Parallel Computing with Dask

Wolfgang Hayek, Maxime Rio, Chris Scott

NZRSE Conference 2020

New Zealand eScience Infrastructure

Overview

1. Dask – Swiss army knife for parallel computing
2. Going HPC - Dask-MPI
3. Bundling things up - Containerisation
4. Summary

Dask – Swiss army knife for parallel computing

Dask

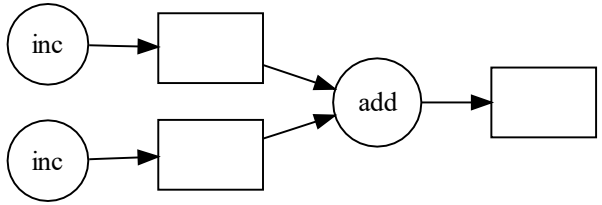
- Python toolbox for parallel processing
- Scalable from laptops to clusters
- Supports multithreading, multiprocessing, MPI, Slurm, …
- Integration with NumPy, Pandas, Scikit-Learn, …
- Low-level APIs “Delayed” and “Futures”

Dask Low-Level APIs

Dask Delayed

```
def inc(x):
    return x+1

def add(x,y):
    return x+y
```




```
x = dask.delayed(inc)(1)
y = dask.delayed(inc)(2)
z = dask.delayed(add)(x, y)
z.compute()
```

- Graphs with parallel execution

Dask Low-Level APIs

Dask Futures

```
from dask.distributed import Client, as_completed

def worker_function(x):
  return f(x)

client = Client()
futures = client.map(worker_function, data)

for future in as_completed(futures):
  write_output(result.result())
```

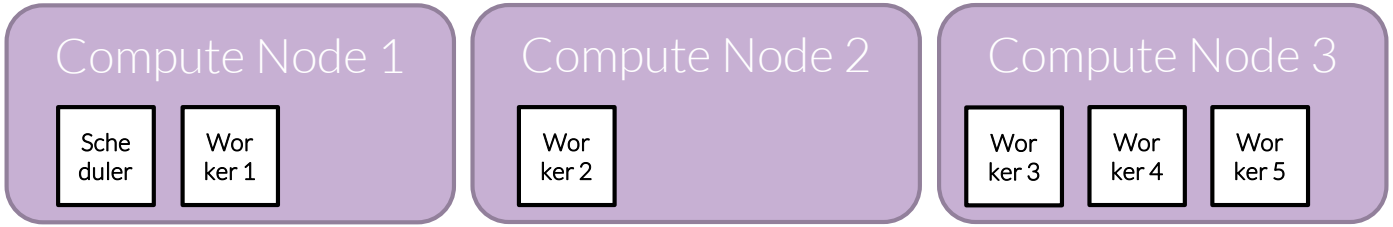
- worker_funct executed on worker processes
- Individual tasks can fail without loosing all results
- API allows fine-grained control

Dask

- Easy to use
- Multitude of APIs cater for a lot of use cases
- Works well for “loosely coupled” workflows that are not affected by latencies/overheads
- Not suitable for tightly coupled simulations (e.g., weather forecasting models)
- Neat browser dashboard to check on workers and progress

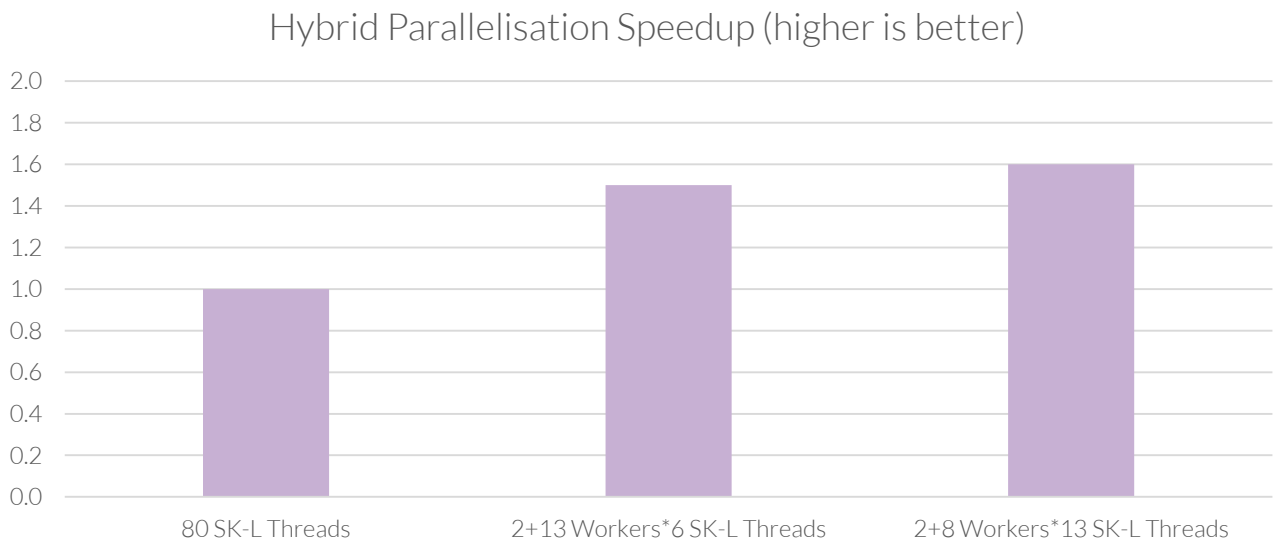
Going HPC – Dask-MPI

Dask-MPI



- Uses MPI instead of Python multiprocessing
- MPI: Message Passing Interface
- Allows scaling up to a cluster
- Requires at least 3 ranks (scheduler + 2 workers)
- Useful for large workloads
- Great throughput by “filling the gaps” on busy HPC

Dask-MPI



- NeSI consultancy project – Dask + SciKit-Learn
- 1 HPC node with 80 logical Skylake cores
- Hybrid Dask+SciKit-Learn parallelisation better than using only SK-L multithreading

Dask-MPI

- Need to use mpi4py with MPICH/Intel MPI
- mpi4py and Dask-MPI easily installed with conda
- Useful for batch jobs

Alternative: Dask-jobqueue (see Maxime's demo)

- Works directly with Slurm
- Useful for interactive jobs

Containerisation

Containerisation

- Dask/Dask-MPI most easily installed via Conda
- Can be part of complex workflows with a large number of dependency packages
- Good candidate for containerisation
- Singularity: HPC-ready with good MPI support

Containerisation

- Container needs to use matching MPI distribution for Dask-MPI (Intel MPI on Mahuika)
- Reduces portability – MPICH-type MPI libraries not compatible with OpenMPI
- Integration with MPI and Slurm is easy:

```
mpirun singularity exec dask-container.sif python parallelprog.py

srun singularity exec dask-container.sif python parallelprog.py
```

- Try it yourself in the Singularity workshop on Friday!

Summary

Summary

- Dask makes parallel programming really easy
- Multitude of APIs and parallel backends should cater for many use cases
- Keep overheads in mind!

If you are interested in more…

- Maxime's Demo on Jupyter+Dask+Slurm
- Singularity workshop with Dask-MPI example

Let us know if you would like help: **support@nesi.org.nz**

Summary

Thank you!